UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/827,042	04/19/2004	Michael G. Martinek	IGT1P369D1/SH-052 DIV	2382

22434 7590 02/12/2007

BEYER WEAVER LLP
P.O. BOX 70250
OAKLAND, CA 94612-0250

EXAMINER	
KARKHANIS, AASHISH	
ART UNIT	**PAPER NUMBER**
3714	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	02/12/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	**Application No.** 10/827,042	**Applicant(s)** MARTINEK ET AL.	
	Examiner Aashish Karkhanis	**Art Unit** 3714	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)□ Responsive to communication(s) filed on *01 June 2006*.

2a)□ This action is **FINAL**. 2b)☒ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *48-63* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *48-63* is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)□ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *19 April 2004* is/are: a)☒ accepted or b)□ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)□ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)□ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)□ All b)□ Some * c)□ None of:

1.□ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____.

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) □ Information Disclosure Statement(s) (PTO/SB/08)
Paper No(s)/Mail Date _____.

4) □ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) □ Notice of Informal Patent Application

6) □ Other: _____.

DETAILED ACTION

Double Patenting

The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. A nonstatutory obviousness-type double patenting rejection is appropriate where the conflicting claims are not identical, but at least one examined application claim is not patentably distinct from the reference claim(s) because the examined application claim is either anticipated by, or would have been obvious over, the reference claim(s). See, e.g., In re Berg, 140 F.3d 1428, 46 USPQ2d 1226 (Fed. Cir. 1998); In re Goodman, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); In re Longi, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); In re Van Ornum, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); In re Vogel, 422 F.2d 438, 164 USPQ 619 (CCPA 1970); and In re Thorington, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

A timely filed terminal disclaimer in compliance with 37 CFR 1.321(c) or 1.321(d) may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent either is shown to be commonly owned with this application, or claims an invention made as a result of activities undertaken within the scope of a joint research agreement.

Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

1. Claims 48-63 are provisionally rejected on the ground of nonstatutory obviousness-type double patenting as being unpatentable over new claims 58-73 of copending Application No. 09/520,405 submitted in an amendment received on December 2, 2004. Although the conflicting claims are not identical, they are not patentably distinct from each other because when claims in one application are so close in content to claims in another application that they both cover the same subject matter, despite a slight different in wording, it is proper to apply obvious-type double patenting. The claims of co-pending Application No. 09/520,405 recite "...a computerized wagering game controller comprising a processor with a memory and an operating system stored

in said memory..." and the claims of the instant application recite "...a universal operating system stored in memory of a computerized controller comprising..." which are considered to be equivalent, despite the slight difference in wording. Thus, claims 48-63 of the instant application are considered to be similar, if not identical, to claims 58-73 of co-pending Application No. 09/520,405.

This is a provisional obviousness-type double patenting rejection because the conflicting claims have not in fact been patented.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. Claims 48-50, 54, 59, 61, & 63 are rejected under 35 U.S.C. 103(a) as being unpatentable over Mathur et al. (U.S. Patent No. 6,671,745) in view of in view of Pascal (U.S. Patent No.5,971,851).

Claims 48, 53 & 59: Mathur discloses the same invention including a universal operating system stored in a memory of a computerized controller, and including the following features:

i. The computerized controller having a processor,.memory, and a nonvolatile storage (figure 2).

ii. An operating system kernel that executes a system handler application, the system handler application operable to dynamically link with at a plurality of gaming program shared objects and load the program shared objects (figures 2 & 3; column 4, lines 7-12, column 6, lines 28-65, & column 9 line 48 – column 10, line 24).

iii. A system handler having an event queue (column 7, lines 6-10 & lines 57-62; column 8, lines 62-65 & column 9, lines 25-26).

iv. A system handler application having an Application Program Interface (API) having functions callable from the program shared object, the API having a plurality of functions callable by and used by at least some of the shared objects (See id)

v. System handler application operable to initiate a program based on data variables stored in the nonvolatile storage the system handler application operable to write data variables to state storage and nonvolatile storage (column 5, lines 1-26 & column 8, lines 36-65). Specifically, Mathur discloses a process management system, which can load and organize applications within the system (column 7, lines 59-60).

vi. An operating system controlled by a general-purpose computer (column 4, lines 22-30).

vii. Nonvolatile storage stores program variables, such that loss of power does not result in loss of the state of the computerized system (column 5, lines 1-26 & column 8, lines 36-65).

viii. A system handler application that loads a first shared object and the first shared object calls up a function from within an API (figure 2, figure 3, & column 10, lines 13-31).

ix. A file storage system for organizing data for retrieval and use (figure 2, element 218).

x. Mathur describes all the features of the claim except causing the execution of a corresponding callback function when a data variable is changed into non-volatile storage. Pascal discloses an analogous operating system for a gaming device wherein callbacks are employed to communicate information between application modules upon the occurrence of certain events (column 1, line 44 – column 2, line 30). In general, callback routines are used in state-based machines to communicate data between independent modules upon the occurrence of predetermined events (column 6, lines 25-45). Pascal describes using callback to enhance the robustness of a gaming device under fault conditions to protect data that may affect the outcome of a game payout (column 2, lines 25-30). In view of Pascal, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the customized gaming operating system suggested by the combination of Mathur with Brunner, wherein the system enhance security by monitoring application modules, to execute a callback function corresponding to a change in game data stored in non-volatile memory to enhance the security of the gaming device by monitoring changes in data that might affect the outcome of the game payout and thereby provide a more secure gaming device that is resistant to errors caused by losses in power or tampering.

Claim 49: Mathur discloses a system handler application comprising an event handler, which handles events (column 8, lines 62-65).

Claim 50: Mathur discloses a system handler unloading, loading and executing program shared objects (column 7, line 56 – column 8, line 15). Specifically, Mathur discloses a process management system, which can load and organize applications within the system (column 7, lines 59-60).

Claim 54: Mathur discloses a plurality of APIs (figures 2 & 3).

Claim 61: Mathur discloses that the execution of the operating system loads and operates the system handler application, the system handler application operable to dynamical link with a plurality of program shared objects and load said shared objects the system handler application having an API having a plurality of functions callable from at least some of the shared objects; the system handler application operable to initiate an software application based on data variables stored in a nonvolatile storage and the system handler application operable to write data variables to the nonvolatile storage; the system handler application then loading a first shared object and providing an API functions called by the first shared object, the system handler application then executing the first shared object (figures 2 & 3, column 2, line 56 – column 3, line 36, column 6, lines 28-65, column 8, lines 35-65, & column 10, lines 17-24).

Claim 63: Mathur discloses a system handler application comprising an event handler, which handles events (column 8, lines 62-65).

2. Claims 51, 52 & 62 are rejected under 35 U.S.C. 103(a) as being unpatentable over Mathur in view of Pascal, as applied to claim 48 above, further in view of Brunner et al. (U.S. Patent No. 4,727,544).

Claims 51: Mathur discloses a system handler that stores data variables modified by program shared objects in non-volatile storage and state storage to ensure data is not lost during a critical failure such as a power loss (column 8, lines 15-46). Mathur does not describe, verifying the code for a shared object has not changed. Gaming regulations require that controllers include mechanisms to verify executable code and data, which may affect payouts or game outcomes. Brunner, for example, teaches that known gaming devices include memory-checking software, which is implemented when a device is powered-up to detect unauthorized memory changes (column 1, lines 13-32). The level (e.g. kernel, operating system or application) at which such software is implemented is within purview of the designer. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the special purpose controller disclosed by Mathur, wherein the system handler executes a program to store application and state data to prevent loss after a power failure, to add the feature of verifying this code for shared objects has not changed in order to meet gaming regulations which require that controllers include mechanisms to verify executable code and data which may affect payouts or game outcomes.

Claim 52: Mathur discloses an index of pointers that associate variable names with data locations (column 7, lines 4-9). Data may be stored in non-volatile memory (column 5, lines 1-41 & column 8, lines 36-46).

Claim 62: The combination of Mathur with Brunner describes all the features of the claim except causing the execution of a corresponding callback function when a data variable is changed into non-volatile storage. Pascal discloses an analogous operating system for a gaming device wherein callbacks are employed to communicate information between application modules upon the occurrence of certain events (column 1, line 44 – column 2, line 30). In general, callback routines are used in state-based machines to communicate data between independent modules upon the occurrence of predetermined events (column 6, lines 25-45). Pascal describes using callback to enhance the robustness of a gaming device under fault conditions to protect data that may affect the outcome of a game payout (column 2, lines 25-30). In view of Pascal, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the customized gaming operating system suggested by the combination of Mathur with Brunner, wherein the system enhance security by monitoring application modules, to execute a callback function corresponding to a change in game data stored in non-volatile memory to enhance the security of the gaming device by monitoring changes in data that might affect the outcome of the game payout and thereby provide a more secure gaming device that is resistant to errors caused by losses in power or tampering.

3. Claims 55-57 & 60 are rejected under 35 U.S.C. 103(a) as being unpatentable over Mathur in view of Pascal as applied to claim 48 above, further in view of Official Notice.

Claim 55: Mathur discloses an operating system, but it is not Linux. The Examiner takes Official Notice that Linux is a well-known, commercially available operating system that is substitutable for the same purpose as the operating system describe by Mathur. It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the controller disclosed by Mathur to substitute the Linux operating system in order to use an open-source, freely modifiable operating system, giving a developer more freedom to customize software to suit the needs of a situation.

Claims 56 and 57: Mathur discloses shared objects including device handlers wherein at least one device handler for a device is disabled, where a device handler is generic and may control many different types of devices (column 3, lines 5-22 & column 9, line 48 – column 10, line 49).

Claim 60: Mathur discloses a system handler application, which loads and executes shared objects wherein the shared objects are operable to share data via program variables stored in non-volatile storage (figure 2, figure 3, & column

5, lines 1-41). Mathur states that different applications require different numbers of shared objects (column 2, lines 18-49). Because of this, Mathur anticipates that any number of shared objects, including one, may be executed at a time, for a range of purposes, including for a system employing large objects that require the controller's full resources.

4. Claim 58 is rejected under 35 U.S.C. 103(a) as being unpatentable over Mathur in view of Pascal and further in view of Official Notice, as applied to claim 57 above, further in view of Angelo (U.S. Patent No. 5,944,821).

Claim 58: Mathur discloses a computerized game controller as described above, but does not disclose a hashing function for an operating system and system handler. Angelo teaches a method of storing secure hash values of operating system programs, programs, which include an operating system and system handler (column 3, line 18). Implementing a hashing function throughout an operating system enhances security within the operating system. Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the computerized game controller of Mathur with the secure hashing method for operating system programs as taught by Angelo in order to ensure the integrity of a computerized game controller and increase security within the operating system.

Response to Arguments

5. Applicant's arguments have been fully considered but they are not persuasive.

Applicant maintains that Mathur and Pascal together do not teach the use of callback functions executed when data in non-volatile memory is modified. The examiner respectfully disagrees. Mathur discloses all elements of claim 48, except for the use of callback functions executed when data in non-volatile memory is modified. Pascal explicitly teaches the use of callback functions between application modules when certain generic events occur (column 1, line 44 – column 2, line 30), but does not teach specifically tracking the events related to change of variables in non-volatile memory. However, because Mathur and Pascal teach the use of callback functions in an application handling system which occur when external events occur in order to provide an efficient mechanism for communication, it would have been obvious to one of ordinary skill in the art at the time of the invention to include an event such as the modification of data in place of a generic event which calls a callback function in order to provide an efficient method of system control.

Applicant also maintains that claim 61 distinguishes over the prior art because both a system handler and kernel link and load shared objects and device handlers. However, examiner states that Mathur does in fact disclose this feature. Examiner returns to the passages of Mathur cited by Applicant. "The core system interface is the module through which applications can access the operating system." Mathur in this passage discloses a system handler which is a loadable kernel module, and therefore works together with a kernel module to produce the desired effects as described above. Modules, as are well known and established in the art of kernel development, are simply loadable extensions that extend a kernel's core functions. However, once they are loaded, they function as an integral part of the kernel until unloaded. In this way, a system handler and kernel work together, and by so doing, both perform the function of linking and loading shared objects and device handlers. Further, Applicant implies that the system handler and kernel both independently perform the above operations, but nowhere is this feature claimed.

For the above reasons, claims 48 – 63 stand rejected.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Aashish Karkhanis whose telephone number is (571) 272-2774. The examiner can normally be reached on 0800-1630 M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Robert Olszewski can be reached on (571) 272-6788. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

ARK

CORBETT B. COBURN
PRIMARY EXAMINER